TABLE 5–1 Example addition instructions.

Addition

Assembly Language	*Operation*
ADD AL,BL	AL = AL + BL
ADD CX,DI	CX = CX + DI
ADD EBP,EAX	EBP = EBP + EAX
ADD CL,44H	CL = CL + 44H
ADD BX,245FH	BX = BX + 245FH
ADD EDX,12345H	EDX = EDX + 12345H
ADD [BX],AL	AL adds to the byte contents of the data segment memory lo addressed by BX with the sum stored in the same memory l
ADD CL,[BP]	The byte contents of the stack segment memory location a by BP add to CL with the sum stored in CL
ADD AL,[EBX]	The byte contents of the data segment memory location ad EBX add to AL with the sum stored in AL
ADD BX,[SI+2]	The word contents of the data segment memory location a SI + 2 add to BX with the sum stored in BX
ADD CL,TEMP	The byte contents of data segment memory location TEMP with the sum stored in CL
ADD BX,TEMP[DI]	The word contents of the data segment memory location a TEMP + DI add to BX with the sum stored in BX
ADD [BX+D],DL	DL adds to the byte contents of the data segment memory lo addressed by BX + DI with the sum stored in the same memo
ADD BYTE PTR [DI],3	A 3 adds to the byte contents of the data segment memory addressed by DI with the sum stored in the same location
ADD BX,[EAX+2*ECX]	The word contents of the data segment memory location a EAX plus 2 times ECX add to BX with the sum stored in BX
ADD RAX,RBX	RBX adds to RAX with the sum stored in RAX (64-bit mode
ADD EDX,[RAX+RCX]	The doubleword in EDX is added to the doubleword addres sum of RAX and RCX and the sum is stored in EDX (64-bit

TABLE 5–2 Example increment instructions. **Increment**

Assembly Language	*Operation*
INC BL	BL = BL + 1
INC SP	SP = SP + 1
INC EAX	EAX = EAX + 1
INC BYTE PTR[BX]	Adds 1 to the byte contents of the data segment memory location addressed by BX
INC WORD PTR[SI]	Adds 1 to the word contents of the data segment memory location addressed by SI
INC DWORD PTR[ECX]	Adds 1 to the doubleword contents of the data segment memory location addressed by ECX
INC DATA1	Adds 1 to the contents of data segment memory location DATA1
INC RCX	Adds 1 to RCX (64-bit mode)

TABLE 5–3 Example add-with-carry instructions.

ADD with Carry

Assembly Language	*Operation*
ADC AL,AH	AL = AL + AH + carry
ADC CX,BX	CX = CX + BX + carry
ADC EBX,EDX	EBX = EBX + EDX + carry
ADC RBX,0	RBX = RBX + 0 + carry (64-bit mode)
ADC DH,[BX]	The byte contents of the data segment memory location addressed by BX add to DH with the sum stored in DH
ADC BX,[BP+2]	The word contents of the stack segment memory location addressed by BP plus 2 add to BX with the sum stored in BX
ADC ECX,[EBX]	The doubleword contents of the data segment memory location addressed by EBX add to ECX with the sum stored in ECX

TABLE 5–4 Example subtraction instructions. **Subtraction**

Assembly Language	*Operation*
SUB CL,BL	CL = CL – BL
SUB AX,SP	AX = AX – SP
SUB ECX,EBP	ECX = ECX – EBP
SUB RDX,R8	RDX = RDX – R8 (64-bit mode)
SUB DH,6FH	DH = DH – 6FH
SUB AX,0CCCCH	AX = AX – 0CCCCH
SUB ESI,2000300H	ESI = ESI – 2000300H
SUB [DI],CH	Subtracts CH from the byte contents of the data segment memory addressed by DI and stores the difference in the same memory location
SUB CH,[BP]	Subtracts the byte contents of the stack segment memory location addressed by BP from CH and stores the difference in CH
SUB AH,TEMP	Subtracts the byte contents of memory location TEMP from AH and stores the difference in AH
SUB DI,TEMP[ESI]	Subtracts the word contents of the data segment memory location addressed by TEMP plus ESI from DI and stores the difference in DI
SUB ECX,DATA1	Subtracts the doubleword contents of memory location DATA1 from ECX and stores the difference in ECX
SUB RCX,16	RCX = RCX – 18 (64-bit mode)

TABLE 5–5 Example decrement instructions. **Decrement**

Assembly Language	*Operation*
DEC BH	BH = BH – 1
DEC CX	CX = CX – 1
DEC EDX	EDX = EDX – 1
DEC R14	R14 = R14 – 1 (64-bit mode)
DEC BYTE PTR[DI]	Subtracts 1 from the byte contents of the data segment memory location addressed by DI
DEC WORD PTR[BP]	Subtracts 1 from the word contents of the stack segment memory location addressed by BP
DEC DWORD PTR[EBX]	Subtracts 1 from the doubleword contents of the data segment memory location addressed by EBX
DEC QWORD PTR[RSI]	Subtracts 1 from the quadword contents of the memory location addressed by RSI (64-bit mode)
DEC NUMB	Subtracts 1 from the contents of data segment memory location NUMB

Subtraction with Borrow

TABLE 5–6 Example subtraction-with-borrow instructions.

Assembly Language	*Operation*
SBB AH,AL	AH = AH – AL – carry
SBB AX,BX	AX = AX – BX – carry
SBB EAX,ECX	EAX = EAX – ECX – carry
SBB CL,2	CL = CL – 2 – carry
SBB RBP,8	RBP = RBP– 2 – carry (64-bit mode)
SBB BYTE PTR[DI],3	Both 3 and carry subtract from the data segment memory location addressed by DI
SBB [DI],AL	Both AL and carry subtract from the data segment memory location addressed by DI
SBB DI,[BP+2]	Both carry and the word contents of the stack segment memory location addressed by BP plus 2 subtract from DI
SBB AL,[EBX+ECX]	Both carry and the byte contents of the data segment memory location addressed by EBX plus ECX subtract from AL

TABLE 5–7 Example comparison instructions.

Comparison

Assembly Language	*Operation*
CMP CL,BL	CL – BL
CMP AX,SP	AX – SP
CMP EBP,ESI	EBP – ESI
CMP RDI,RSI	RDI – RSI (64-bit mode)
CMP AX,2000H	AX – 2000H
CMP R10W,12H	R10 (word portion) – 12H (64-bit mode)
CMP [DI],CH	CH subtracts from the byte contents of the data segment memory location addressed by DI
CMP CL,[BP]	The byte contents of the stack segment memory location addressed by BP subtracts from CL
CMP AH,TEMP	The byte contents of data segment memory location TEMP subtracts from AH
CMP DI,TEMP[BX]	The word contents of the data segment memory location addressed by TEMP plus BX subtracts from DI
CMP AL,[EDI+ESI]	The byte contents of the data segment memory location addressed by EDI plus ESI subtracts from AL

Multiplication

TABLE 5–8 Example 8-bit multiplication instructions.

Assembly Language	*Operation*
MUL CL	AL is multiplied by CL; the unsigned product is in AX
IMUL DH	AL is multiplied by DH; the signed product is in AX
IMUL BYTE PTR[BX]	AL is multiplied by the byte contents of the data segment memory location addressed by BX; the signed product is in AX
MUL TEMP	AL is multiplied by the byte contents of data segment memory location TEMP; the unsigned product is in AX

TABLE 5–9 Example 16-bit multiplication instructions.

Assembly Language	*Operation*
MUL CX	AX is multiplied by CX; the unsigned product is in DX–AX
IMUL DI	AX is multiplied by DI; the signed product is in DX–AX
MUL WORD PTR[SI]	AX is multiplied by the word contents of the data segment memory location addressed by SI; the unsigned product is in DX–AX

TABLE 5–10 Example 32-bit multiplication instructions.

Assembly Language	*Operation*
MUL ECX	EAX is multiplied by ECX; the unsigned product is in EDX–EAX
IMUL EDI	EAX is multiplied by EDI; the signed product is in EDX–EAX
MUL DWORD PTR[ESI]	EAX is multiplied by the doubleword contents of the data segment memory location address by ESI; the unsigned product is in EDX–EAX

TABLE 5–11 Example 64-bit multiplication instructions.

Assembly Language	*Operation*
MUL RCX	RAX is multiplied by RCX; the unsigned product is in RDX–RAX
IMUL RDI	RAX is multiplied by RDI; the signed product is in RDX–RAX
MUL QWORD PTR[RSI]	RAX is multiplied by the quadword contents of the memory location address by RSI; the unsigned product is in RDX–RAX

Division

TABLE 5–12 Example 8-bit division instructions.

Assembly Language	*Operation*
DIV CL	AX is divided by CL; the unsigned quotient is in AL and the unsigned remainder is in AH
IDIV BL	AX is divided by BL; the signed quotient is in AL and the signed remainder is in AH
DIV BYTE PTR[BP]	AX is divided by the byte contents of the stack segment memory location addressed by BP; the unsigned quotient is in AL and the unsigned remainder is in AH

TABLE 5–13 Example 16-bit division instructions.

Assembly Language	*Operation*
DIV CX	DX–AX is divided by CX; the unsigned quotient is AX and the unsigned remainder is in DX
IDIV SI	DX–AX is divided by SI; the signed quotient is in AX and the signed remainder is in DX
DIV NUMB	DX–AX is divided by the word contents of data segment memory NUMB; the unsigned quotient is in AX and the unsigned remainder is in DX

TABLE 5–14 Example 32-bit division instructions.

Assembly Language	*Operation*
DIV ECX	EDX–EAX is divided by ECX; the unsigned quotient is in EAX and the unsigned remainder is in EDX
IDIV DATA4	EDX–EAX is divided by the doubleword contents in data segment memory location DATA4; the signed quotient is in EAX and the signed remainder is in EDX
DIV DWORD PTR[EDI]	EDX–EAX is divided by the doubleword contents of the data segment memory location addressed by EDI; the unsigned quotient is in EAX and the unsigned remainder is in EDX

TABLE 5–15 Example 64-bit division instructions.

Assembly Language	*Operation*
DIV RCX	RDX–RAX is divided by RCX; the unsigned quotient is in RAX and the unsigned remainder is in RDX
IDIV DATA4	RDX–RAX is divided by the quadword contents in memory location DATA4; the signed quotient is in RAX and the signed remainder is in RDX
DIV QWORD PTR[RDI]	RDX–RAX is divided by the quadword contents of the memory location addressed by RDI; the unsigned quotient is in RAX and the unsigned remainder is in RDX

TABLE 5–16 Example AND instructions.

AND

Assembly Language	*Operation*
AND AL,BL	AL = AL and BL
AND CX,DX	CX = CX and DX
AND ECX,EDI	ECX = ECX and EDI
AND RDX,RBP	RDX = RDX and RBP (64-bit mode)
AND CL,33H	CL = CL and 33H
AND DI,4FFFH	DI = DI and 4FFFH
AND ESI,34H	ESI = ESI and 34H
AND RAX,1	RAX = RAX and 1 (64-bit mode)
AND AX,[DI]	The word contents of the data segment memory location addressed by DI are ANDed with AX
AND ARRAY[SI],AL	The byte contents of the data segment memory location addressed by ARRAY plus SI are ANDed with AL
AND [EAX],CL	CL is ANDed with the byte contents of the data segment memory location addressed by ECX

FIGURE 5–4 The operation of the AND function showing how bits of a number are cleared to zero.

```
  x x x x  x x x x   Unknown number
• 0 0 0 0  1 1 1 1   Mask
-------------------
  0 0 0 0  x x x x   Result
```

TABLE 5–17 Example OR instructions.

Assembly Language	*Operation*
OR AH,BL	AL = AL or BL
OR SI,DX	SI = SI or DX
OR EAX,EBX	EAX = EAX or EBX
OR R9,R10	R9 = R9 or R10 (64-bit mode)
OR DH,0A3H	DH = DH or 0A3H
OR SP,990DH	SP = SP or 990DH
OR EBP,10	EBP = EBP or 10
OR RBP,1000H	RBP = RBP or 1000H (64-bit mode)
OR DX,[BX]	DX is ORed with the word contents of data segment memory location addressed by BX
OR DATES[DI + 2],AL	The byte contents of the data segment memory location addressed by DI plus 2 are ORed with AL

FIGURE 5–6 The operation of the OR function showing how bits of a number are set to one.

```
  x x x x  x x x x   Unknown number
+ 0 0 0 0  1 1 1 1   Mask
-------------------
  x x x x  1 1 1 1   Result
```

TABLE 5–18 Example Exclusive-OR instructions.

XOR

Assembly Language	*Operation*
XOR CH,DL	CH = CH xor DL
XOR SI,BX	SI = SI xor BX
XOR EBX,EDI	EBX = EBX xor EDI
XOR RAX,RBX	RAX = RAX xor RBX (64-bit mode)
XOR AH,0EEH	AH = AH xor 0EEH
XOR DI,00DDH	DI = DI xor 00DDH
XOR ESI,100	ESI = ESI xor 100
XOR R12,20	R12 = R12 xor 20 (64-bit mode)
XOR DX,[SI]	DX is Exclusive-ORed with the word contents of the data segment memory location addressed by SI
XOR DEAL[BP+2],AH	AH is Exclusive-ORed with the byte contents of the stack segment memory location addressed by BP plus 2

FIGURE 5–8 The operation of the Exclusive-OR function showing how bits of a number are inverted.

x x x x x x x x	Unknown number
⊕ 0 0 0 0 1 1 1 1	Mask
x x x x $\overline{x}$ $\overline{x}$ $\overline{x}$ $\overline{x}$	Result

TEST

Assembly Language	*Operation*
TEST DL,DH	DL is ANDed with DH
TEST CX,BX	CX is ANDed with BX
TEST EDX,ECX	EDX is ANDed with ECX
TEST RDX,R15	RDX is ANDed with R15 (64-bit mode)
TEST AH,4	AH is ANDed with 4
TEST EAX,256	EAX is ANDed with 256

TABLE 5–20 Bit test instructions.

Assembly Language	*Operation*
BT	Tests a bit in the destination operand specified by the source operand
BTC	Tests and complements a bit in the destination operand specified by the source operand
BTR	Tests and resets a bit in the destination operand specified by the source operand
BTS	Tests and sets a bit in the destination operand specified by the source operand

TABLE 5–21 Example NOT and NEG instructions. NOT & NEG

Assembly Language	*Operation*
NOT CH	CH is one's complemented
NEG CH	CH is two's complemented
NEG AX	AX is two's complemented
NOT EBX	EBX is one's complemented
NEG ECX	ECX is two's complemented
NOT RAX	RAX is one's complemented (64-bit mode)
NOT TEMP	The contents of data segment memory location TEMP is one's complemented
NOT BYTE PTR[BX]	The byte contents of the data segment memory location addressed by BX are one's complemented

TABLE 5–22 Example shift instructions.

Shift

Assembly Language	*Operation*
SHL AX,1	AX is logically shifted left 1 place
SHR BX,12	BX is logically shifted right 12 places
SHR ECX,10	ECX is logically shifted right 10 places
SHL RAX,50	RAX is logically shifted left 50 places (64-bit mode)
SAL DATA1,CL	The contents of data segment memory location DATA1 are arithmetically shifted left the number of spaces specified by CL
SHR RAX,CL	RAX is logically shifted right the number of spaces specified by CL (64-bit mode)
SAR SI,2	SI is arithmetically shifted right 2 places
SAR EDX,14	EDX is arithmetically shifted right 14 places

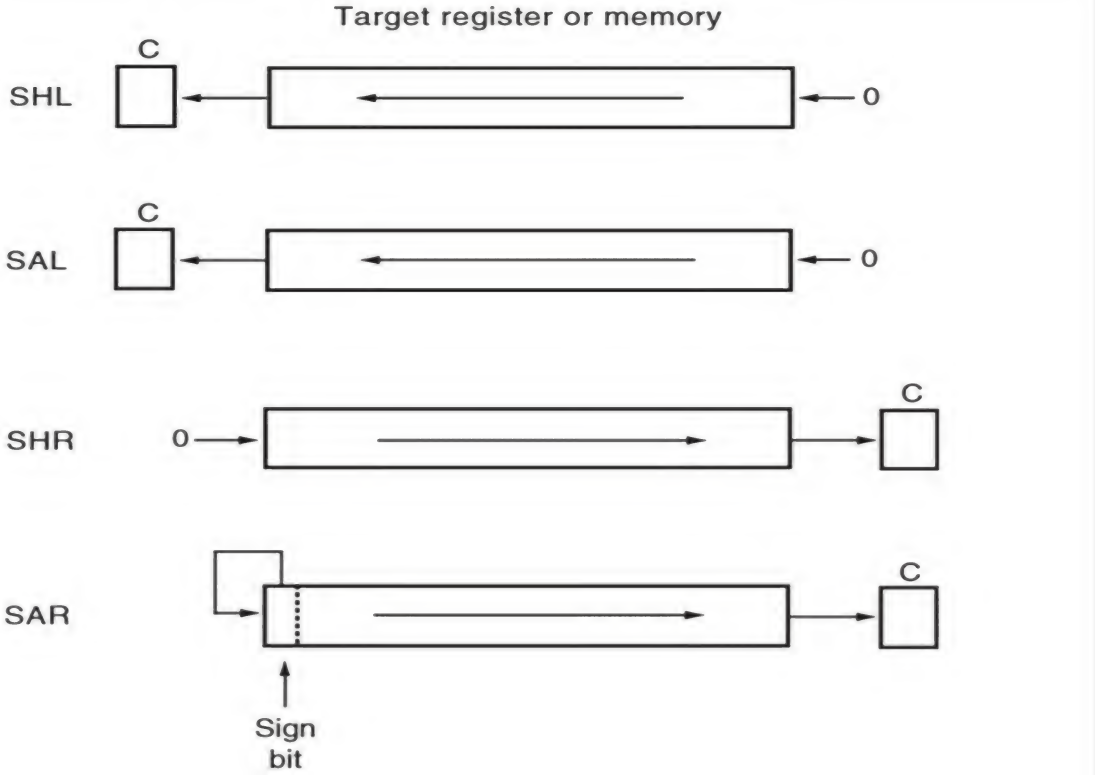
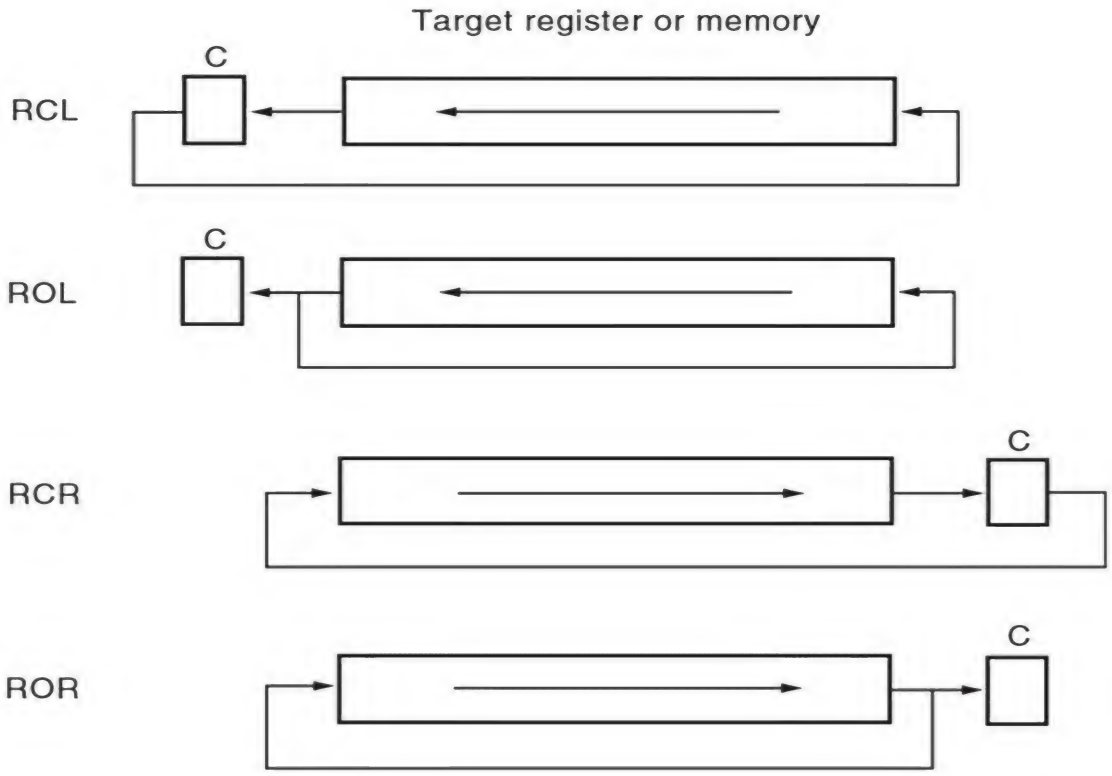


TABLE 5–23 Example rotate instructions.

Rotate

Assembly Language	*Operation*
ROL SI,14	SI rotates left 14 places
RCL BL,6	BL rotates left through carry 6 places
ROL ECX,18	ECX rotates left 18 places
ROL RDX,40	RDX rotates left 40 places
RCR AH,CL	AH rotates right through carry the number of places specified by CL
ROR WORD PTR[BP],2	The word contents of the stack segment memory location addressed by BP rotate right 2 places